

日本国特許庁  
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出願年月日

Date of Application:

2002年 9月27日

出願番号

Application Number:

特願2002-282977

[ST.10/C]:

[JP2002-282977]

出願人

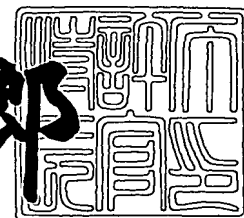
Applicant(s):

ヤマハ株式会社

2003年 6月23日

特許庁長官  
Commissioner,  
Japan Patent Office

太田信一郎



出証番号 出証特2003-3048480

【書類名】 特許願

【整理番号】 20020090

【提出日】 平成14年 9月27日

【あて先】 特許庁長官 殿

【国際特許分類】 G10L 19/06

【発明者】

【住所又は居所】 静岡県浜松市中沢町10番1号  
ヤマハ株式会社内

【氏名】 松沼 康宏

【特許出願人】

【識別番号】 000004075

【氏名又は名称】 ヤマハ株式会社

【代理人】

【識別番号】 100084548

【弁理士】

【氏名又は名称】 小森 久夫

【手数料の表示】

【予納台帳番号】 013550

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9001567

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 オーディオデータ処理装置

【特許請求の範囲】

【請求項 1】 PCMオーディオデータのエンコード処理およびその他の処理  
を実行する複数のプロセッサと、

PCMオーディオデータを複数の分割データに分割する分割手段と、

全ての分割データがエンコードされたとき、エンコードされた各分割データを  
結合して 1 つのデータに再結合する結合手段と、

前記複数のプロセッサの動作状態を監視し、処理を実行していない処理なしの  
プロセッサが生じたとき、このプロセッサに対して前記分割データを供給してエ  
ンコード処理を行わせる管理手段と、

を備えたオーディオデータ処理装置であって、

前記分割手段は、各分割データにその前後の分割データとオーバーラップする  
オーバーラップ区間を持たせて分割し、前記結合手段は、エンコードされた各分  
割データが重なり合うことなく再結合できる結合箇所を、各分割データの前記オ  
ーバーラップ区間のなかで決定するオーディオデータ処理装置。

【請求項 2】 前記管理手段は、前記その他の処理の要求が発生したときに処  
理なしのプロセッサが無かったとき、前記エンコード処理を実行しているプロセ  
ッサの処理を中止させて、このプロセッサに前記その他の処理を実行させる請求  
項 1 に記載のオーディオデータ処理装置。

【請求項 3】 前記複数のプロセッサが実行するその他の処理は、エンコード  
されたデータのデコード処理である請求項 1 または請求項 2 に記載のオーディオ  
データ処理装置。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

この発明は、PCMオーディオデータをMP3などの形式にエンコードするオ  
ーディオデータ処理装置に関する。

【0002】

## 【発明が解決しようとする課題】

PCMオーディオデータを高品質のMP3データにエンコードする処理は、相応の時間を要する処理であり、プロセッサがこの処理に占有されている間は他の処理を実行することができない。このためエンコード処理の途中で他の処理の要求が発生しても、エンコード処理が終了するまでこれに対応することができなかった。さらに、従来は曲データ（PCMオーディオデータ）をMP3データにエンコードする場合、1曲分のデータをそのままエンコード対象としていたため、1回のエンコード処理に要する時間が長く、上記他処理の待ち時間が長くなってしまうという問題点があった。

## 【0003】

また、MP3のエンコード処理は、エンコードするデータの前後にあるデータも参照しながらエンコードを行ってゆくため、エンコード処理が途中で中止されると、それまでのエンコード処理が無駄になり、再度最初からエンコード処理をやりなおさなければならなかった。このため、他の処理の要求に応じてエンコード処理を途中で中止すると、それまでのエンコード処理が無駄になってしまうという問題点があった。

## 【0004】

また、複数のプロセッサを用いれば、MP3エンコード処理を効率的に行うことができるが、各プロセッサに別々の曲データを1曲分ずつエンコード処理させる方式では、曲データのエンコードが終了するまで他処理要求に応じられないという上記課題が解決されない。また、複数のプロセッサをパイプライン接続して、パイプライン処理を行わせれば、1曲分のエンコード処理時間は大幅に短縮されるが、全てのプロセッサがエンコード処理に占有されるため、他処理の要求が発生したとき、一部のプロセッサにこの処理を実行させるという動的な割り当てができないという欠点がある。

## 【0005】

この発明は、オーディオデータのエンコード処理と他の処理とを動的に割り当てながら、効率的にオーディオデータのエンコード処理を行うことができるオーディオデータ処理装置を提供することを目的とする。

## 【 0 0 0 6 】

## 【課題を解決するための手段】

この発明は、PCMオーディオデータのエンコード処理およびその他の処理を実行する複数のプロセッサと、PCMオーディオデータを複数の分割データに分割する分割手段と、全ての分割データがエンコードされたときエンコードされた各分割データを結合して1つのデータに再結合する結合手段と、前記複数のプロセッサの動作状態を監視し、処理を実行していない処理なしのプロセッサが生じたとき、このプロセッサに対して前記分割データを供給してエンコード処理を行わせる管理手段と、を備えたオーディオデータ処理装置であって、

前記分割手段は、各分割データにその前後の分割データとオーバーラップするオーバーラップ区間を持たせて分割し、前記結合手段は、エンコードされた各分割データが重なり合うことなく再結合できる結合箇所を、各分割データの前記オーバーラップ区間のなかで決定することを特徴とする。

## 【 0 0 0 7 】

この発明では、PCMオーディオデータを分割して複数の分割データとし、複数のプロセッサのうち処理なしとなっているプロセッサがあったとき、その空き時間を利用して各分割データ毎のエンコードを行う。このように、複数のプロセッサの空き時間を利用することにより、プロセッサをエンコード処理とその他の処理とに併用することができ、プロセッサの効率的な利用およびエンコード処理の効率化を図ることができ、且つ、分割されたデータは1曲全体のデータに比べて（当然に）短いため、その他の処理の要求が発生した場合でも、待ち時間を短くすることができる。

## 【 0 0 0 8 】

ただし、上述したように、PCMオーディオデータからMP3へエンコードするとき、エンコードするデータだけでなくその前後のデータも参照してエンコード処理を行うため、分割データの両端をエンコードした場合、分割しないでエンコードした場合とではエンコードされたデータの内容が異なってしまう。そこで、各分割データに互いに重なり合う部分（オーバーラップ）を持たせてPCMオーディオデータを分割し、この分割データを別々にMP3エンコードする。そし

て、エンコードしたのち、オーバーラップ区間を重ね合わせて端のデータを捨てて再結合し、データを分割しないでエンコードした場合と同じデータを作成する。これにより、分割して処理した場合でも連続して処理した場合と同様のエンコードデータを得ることができるとともに、1回のエンコード処理を小さく分けることができるため、他の処理の要求が発生した場合でもその待ち時間を短くすることができる。

## 【 0 0 0 9 】

この発明は、上記発明において、前記管理手段は、前記その他の処理の要求が発生したときに処理なしのプロセッサが無かったとき、前記エンコード処理を実行しているプロセッサの処理を中止させて、このプロセッサに前記その他の処理を実行させることを特徴とする。

## 【 0 0 1 0 】

この発明では、その他の処理をエンコード処理よりも優先させている。すなわち、エンコード処理途中にその他の処理の要求が発生した場合には、エンコード処理を中止して、プロセッサにその他の処理を実行させるようにしている。これにより、「その他の処理」を実行するために設けられたプロセッサを利用し、その他の処理の実行の邪魔にならないように空き時間を利用してエンコード処理をすることができ、プロセッサの利用効率を向上させることができる。

なお、エンコード処理を中止しても分割データは短いため処理時間の無駄は僅かである。また、エンコード途中のデータについてエンコード処理が終了した部分と未処理の部分とを再度分割するようにしてもよい。

## 【 0 0 1 1 】

この発明は、上記発明において、前記複数のプロセッサが実行するその他の処理は、エンコードされたデータのデコード処理であることを特徴とする。

## 【 0 0 1 2 】

この発明では、エンコードされたオーディオデータをデコードするプロセッサを利用してPCMオーディオデータのエンコードをすることにより、プロセッサのリソースを効率的に利用することができる。

## 【 0 0 1 3 】

## 【発明の実施の形態】

図面を参照してこの発明の実施形態について説明する。図1はこの発明の実施形態であるオーディオサーバ装置のブロック図である。オーディオサーバ装置は、それぞれ別の部屋に設置されている複数のクライアント装置（不図示）からの再生要求に基づいてオーディオ信号を一括して供給する装置であり、各クライアント装置からリクエストされた曲を各クライアント装置に対して個別に配信する。各クライアント装置は、オーディオサーバ装置に対するリクエスト機能およびアナログオーディオ信号の再生機能を備えた装置である。

## 【0014】

オーディオサーバ装置は、クライアント装置から曲がリクエストされると、このリクエストされた曲のMP3データをHDD11から読み出し、プロセッサであるDSP15でデコードするとともにアナログ回路17でアナログ信号に変換してリクエストのあったクライアント装置に配信する。

## 【0015】

複数のクライアント装置からのリクエストに対応するためにDSP15は複数設けられている。DSP15の個数は、各クライアント装置の稼働率を勘案して、クライアント装置の台数よりも少ない個数（ $n$ 個）設けられる。一方、DAコンバータ・アンプを含むアナログ回路17は、クライアント装置の台数の同数設けられ、それぞれクライアント装置と1対1に対応している。これら複数のDSP15とアナログ回路17はパッチベイ16によって接続される。パッチベイ16は、リクエストのあったクライアント装置に接続されるアナログ回路17とそのリクエストされた曲のMP3データをデコードするDSP15とを接続する。

## 【0016】

このオーディオサーバ装置を制御するCPU10は、図示しない通信機能により各クライアント装置からリクエストを受信する。CPU10が、リクエストを受信すると、DSP15（15-1～15- $n$ ）のうちの1つをこのリクエストの曲データ（MP3データ）をデコードするDSPに割り当て、このDSPとリクエストのあったクライアント装置のアナログ回路17とを接続するようにパッチベイ16を制御する。

## 【 0 0 1 7 】

クライアント装置からのリクエストに対応できるように、HDD 1 1 には複数の曲のMP 3 データが記憶されている。このオーディオサーバ装置は、CD-R OMドライブ 1 2 にセットされたオーディオCDからPCMオーディオデータを読み込んでMP 3 データにエンコードし、このエンコードしたMP 3 データをHDD 1 1 に蓄積記憶する。このエンコード処理は、上記DSP 1 5 ( 1 5 - 1 ~ n ) が行う。すなわち、DSP 1 5 ( 1 5 - 1 ~ n ) のなかに、MP 3 データのデコード処理をしていない空きのDSPがあるとき、このDSPの空き時間を利用して、上記PCMオーディオデータからMP 3 データへのエンコードを行わせるようにしている。

## 【 0 0 1 8 】

DSP 1 5 - 1 ~ n の各DSPのステータス（動作状態）を管理するために、HDD 1 1 上にプロセッサ管理テーブルが設定されている。CPU 1 0 は、各DSPの動作状態を監視し、この処理状態に応じて「他処理中」、「符号化処理中」、「処理なし（空き）」のいずれかのステータス情報をプロセッサ管理テーブルに書き込む。「他処理中」は、MP 3 データのデコード処理中であることを示し、「符号化処理中」は、PCMオーディオデータのエンコード処理中であることを示し、「処理なし」は、いずれの処理もしておらず動作指示待ちであることを示している。

## 【 0 0 1 9 】

また、DSPが1曲分のPCMオーディオデータをエンコードするには相応の時間を要する。もし、エンコード処理中にリクエストがあった場合に、1曲分のPCMオーディオデータ全部のエンコードが終了するまでリクエストを待ち状態にしておくと、クライアント装置の待ち時間が長くなってクライアント装置の稼働効率の低下、顧客に対するサービスの低下を招くことになる。そこで、このオーディオサーバ装置では、1曲分のPCMオーディオデータを複数の分割データに分割し、各分割データ毎に別々にエンコード処理するようにしている。このように、1回の処理量を少なくして処理時間を短くすることにより、全てのDSPが処理を実行中で「処理なし」のDSPがないときにリクエストが発生した場合



であっても、そのリクエストの待ち時間を短くすることができる。

#### 【0020】

曲データを複数の分割データに分割したとき、HDD 1 1 上に分割データ管理テーブルが作成され、各分割データの曲中の位置、大きさ、エンコード済か否かなどが記憶される。分割データ管理テーブルでは、各分割データにレコードが作成され各レコードには、その分割データの先頭フレーム番号、サイズ（フレーム数）、ステータスのデータが含まれている。先頭フレーム番号は、この分割データの先頭フレームが、曲全体の何番目のフレームに当たるかの番号であり、これでこの分割データの位置を知ることができる。サイズはフレーム数で表される。ステータスは、この分割データの処理状態を表す情報であり「未処理」、「符号化中」、「符号化完了」の3つのステータスを持っている。「未処理」は、エンコード処理が開始されていない状態を示し、「符号化中」は現在DSPでMP3エンコード処理が行われている状態を示し、「符号化完了」はエンコード処理が完了した状態を示している。

#### 【0021】

CPU 1 0 は、PCMオーディオデータである曲データを複数の分割データに分割するとともに、MP3にエンコードされた複数の分割データを再結合し、1曲のMP3データに編集する。この1曲に編集されたMP3データが、リクエストに応じて再生されるデータとしてHDD 1 1 に記憶される。

#### 【0022】

ここで、PCMオーディオデータの分割および結合の方式について説明する。オーディオデータの分割は、MP3のフレームサイズを単位として行われ、各分割データには、前後の分割データと数フレームずつオーバーラップするオーバーラップ区間が設けられる。MP3にエンコードされた各分割データは、オーバーラップ区間の解析により、どのフレームで前後の分割データと結合するかが決定される。この決定された結合フレームで前後の分割データを結合（順番に接続）することにより、MP3にエンコードされた形式でもとの1曲のオーディオデータが復元される。

#### 【0023】

図2はMP3データのデータ構成を示す図である。MP3データは、PCMオーディオデータの1152サンプルで1フレームを構成する。各フレームの構成は同図(b)に示すように、ヘッダ、サイドインフォメーション、メインデータなどからなり、ヘッダ内の情報のうち、サンプルレート、ビットレート、パディングの有無の情報によりフレームサイズを割り出すことができる。すなわち、1フレームの大きさ(byte数)は、 $144 \times (\text{ビットレート}) \div (\text{サンプルレート})$ で決定される。たとえば、ビットレート: 128 kbps、サンプルレート(サンプリング周波数fs): 44.1 kHzの場合;

$$144 \times 128000 / 44100 = 417 \text{ byte}$$

であり、小数点以下は切り捨てられるが、MP3データ全体としての伝送ビットレートを合わせるために数フレームに1つの割合で1 byteを加算(パディング)したフレームが構成される。数フレームに1つの割合で418バイトのフレーム(パディングフレーム)を作成する。

【0024】

また、サイドインフォメーションには、メインデータビギンというデータが書き込まれている。このデータは、MP3エンコードされた上記1152サンプルの符号化データ部(メインデータ)がどこから始まるかを示すデータである。ここで、MP3フォーマットの1フレームは、上記のように1152サンプルを対象としたものであるが、この1152サンプル分の符号化データ部(メインデータ)を、このフレーム以外の複数の隣接フレームのメインデータエリアにまたがって配置することを許容している。すなわち、1152サンプルのPCMデータをエンコードしたときのデータサイズはそのPCMデータの状態に応じて変化させることができる。これにより、単調な音の区間ではデータ量を少なく、複雑で変化に富んだ区間にはデータ量を多く割り振ることで、音質を考慮したデータ配分を行うことができる。また、このとき生じるフレーム毎のデータ配分量の差異は、複数の隣接フレーム間でそれぞれのメインデータサイズを調節することで吸収し、データ量が多い区間に多くのメインデータエリアを配分できるようにする。その結果として、データ量の少ない区間では、メインデータエリアの後ろの部分に余白ができることで、次のフレーム以降のデータ量の多い区間により多くの

データを配分するための余裕分が生じることになる（ビット貯蔵）。

【 0 0 2 5 】

そして、次のフレームのメインデータは、そのフレームのメインデータエリアからでなく、このビット貯蔵されている前のフレームのメインデータエリアの途中から書きはじめる。これによって、配信のビットレートを変更することがなくなるうえ、その分のメインデータエリアのエリアを節約している。そして、なかには、エンコードしたメインデータのサイズが1フレームのメインデータエリアの大きさよりも大きい場合があるが、上記のように他のフレームでビット貯蔵しているため、このようなデータを書き込んでも、全体としてフレーム数に対応したメインデータエリアのデータ量に対応するデータ量に収めることができる。

【 0 0 2 6 】

図2（C）にビット貯蔵の例を示す。これは曲の途中のフレームを示す図であり、フレーム(1)のメインデータエリアにはフレーム(2)のメインデータの後半部、フレーム(3)のメインデータおよびフレーム(4)のメインデータの先頭部分が書き込まれている。そして、フレーム(4)のメインデータの残りの部分は、フレーム(2)のメインデータエリアに書き込まれている。このように、メインデータがフレームの範囲を越えて書き込まれるため、PCMオーディオデータの内容に応じて、ビット数を節約できるフレームではメインデータを小さくしてビット貯蔵し、複雑なPCMデータのフレームでは、貯蔵されているビットを借りて1フレームを超える大きさのデータを書き込むことができ、全体としてのデータ量を増加させることなく高音質のエンコードをすることができる。

【 0 0 2 7 】

MP3エンコードにおいては、処理対象フレームとその前後の隣接フレームの半分にわたったフィルタ処理を行う。このため、データを分割してしまうと、その分割点付近のフレームでは隣接フレームの有無による影響が生じてしまう。また、MP3データにはビット貯蔵によるメインデータの移動があるため、オーディオデータを時間軸上でオーバーラップ無しに切断した分割データを単純に結合するのみでは再結合されたデータが不連続になってしまう。

そこで、PCMオーディオデータを分割するときに前後の分割データに重なり

合うオーバーラップ区間を設け、MP3にエンコードしたのち、このオーバーラップ区間内で最適な結合フレームを割り出し、メインデータの連続性を維持しつつ前記割り出された結合フレームで分割データを結合（連結）するようにしている。

## 【0028】

図3は、PCMオーディオデータの分割処理を説明する図である。オーディオデータを分割する場合、各分割データがMP3の1フレームのサンプル数（この実施形態では1152サンプル）の整数倍になるように分割する。さらに、各分割データが前後の分割データと数フレーム分のオーバーラップ区間を共有するように、このオーバーラップ区間を両方に重複させて分割データを作成する。オーバーラップ区間のフレーム数は、上記のフィルタ処理に必要な隣接フレームをカバーできるフレーム数およびビット貯蔵が適合するフレームを検索するためのフレーム数を加算したフレーム数とする。

## 【0029】

同図では、1曲のオーディオデータを4分割し、各分割データが均等に同じ長さになるように調節している。このため、各分割データの長さ（フレーム数）を  $base + ovl$  のフレーム数にしている。なお、基本分割フレーム数  $base$  は、総データフレーム数；  $size$ 、分割数；  $N$ 、オーバーラップフレーム数；  $ovl$  とすると、  $(size - ovl) / N$  で求めることができる。したがって、最初の分割データおよび最後の分割データは、隣の分割データとオーバーラップするのは片方のみであるため、両方ともオーバーラップする中間の分割データに比べてオーバーラップしない区間が  $ovl / 2$  だけ長くなっている。

## 【0030】

なお、元々のPCMオーディオデータのデータ長（サンプル数）がMP3の1フレームのサンプル数（1152サンプル）の整数倍でない場合には、最後の1フレームは短いフレームとなる。また、PCMオーディオデータのフレーム数が分割数  $N$  で割り切れる数でなかった場合には、最後（ $N$  番目）の分割データは他の分割データよりも短い分割データとなる。上記のように分割された各分割データはHDD11に仮記憶され、DSP15-1～ $n$ のいずれかによってMP3デ

ータにエンコードされる。DSP15-1~nは、クライアント装置のリクエストに応じたデコード処理の合間に上記分割データをエンコード処理する。

#### 【0031】

1曲分の複数の分割データが全てMP3データにエンコードされると、このMP3エンコードされた分割データを結合して1曲のMP3データに再編集する。図4および図5は、結合フレーム検索処理を説明する図である。図4(A)は、分割データMP3(1)と分割データMP3(2)とを結合する場合について説明している。分割データMP3(1)のうちオーバーラップ区間の最後の2フレームは、終了点差などの影響を受ける区間であるため、それ以前のフレームの品質を確保するためのダミーフレームとして捨てることとする。また同様に、分割データMP3(2)のうちオーバーラップ区間の最初の2フレームは、フィルタリング遅延や開始点差などの影響を受ける区間であるため、それ以後のフレームの品質を確保するためのダミーフレームとして捨てることとする。したがって、オーバーラップ区間のうち両端の2フレームずつを除いた中央のフレームのどれかを結合フレームとするため、この区間の分割データMP3(1)、分割データMP3(2)の対応する各フレームのメインデータビギン(ビット貯蔵値)の適合性をチェックする。

#### 【0032】

すなわち、同図(B)に示すように、前のデータである分割データMP3(1)のメインデータビギンが、後のデータである分割データMP3(2)のメインデータビギンと同じか、または、それよりも手前にあれば、分割データMP3(1)のメインデータと分割データMP3(2)のメインデータを重なり合うことなく結合することができる。

#### 【0033】

なお、ビット貯蔵に応じて複数の隣接フレーム間でデータ配分を調節するときには、エンコードを行うPCMデータの聴覚的情報量とその時点のビット貯蔵値によって配分量を決定する。このため、DSP15-1~nがそれぞれ別々に実行するMP3エンコード処理を同じものとし、オーバーラップ領域について同じPCMデータに対してエンコードを行ったとしても、オーバーラップ領域の始端か

らエンコードを開始したものと、その前からエンコードをしてきたものとは、オーバーラップ領域に至るまでのエンコード過程が異なるため、それぞれのビット貯蔵値が異なる。したがって、結果として、それぞれのオーバーラップ領域のビット貯蔵値とメインデータビギン値は異なるものになる。このため、図4（A）に示すオーバーラップ領域のうち両端を除く全域にわたってメインデータビギンを比較し、メインデータが重ならない範囲でMP 3（1）のメインデータビギンとMP 3（2）のメインデータビギンが近いフレームを検索して、それを結合フレームとする。

## 【0034】

メインデータが重ならない範囲で、MP 3（1）のメインデータビギンとMP 3（2）のメインデータビギンが近いフレームで両分割データを結合すると、図4（B）の下段に示す結合後のメインデータのマージン（空白エリア）が少なくなり、空白を埋めるためのダミーデータを少なくしてメインデータエリアを有効に使うことができる。

## 【0035】

図5は、CPU 10が実行する結合フレーム検索動作を示すフローチャートである。まず、s 1で検索範囲およびレジスタのリセットを行う。結合する分割データMP 3（1）、MP 3（2）のオーバーラップ区間のうち、両端のデータを除いた検索範囲の先頭フレームの番号をそれぞれi、jにセットし、検索範囲の最終フレームの番号をそれぞれend\_\_i、end\_\_jにセットする。また、メインデータビギンの差が最小のフレーム番号を記憶するレジスタmin\_\_i、min\_\_jにダミーデータとして-1をセットする。

## 【0036】

以下、検索範囲の先頭フレームから最終フレームまでメインデータビギンを比較してゆく。分割データMP 3（1）のフレームiのメインデータビギンを読み出してレジスタAに書き込み（s 2）、分割データMP 3（2）のフレームjのメインデータビギンを読み出してレジスタBに書き込む（s 3）。このレジスタA、Bの内容を比較して結合条件（ $A \geq B$ ）を満たし（s 4）、且つ、その差（ $A - B$ ）が今まで比較したフレームのうち最も小さい場合には（s 5）、これを結合

フレームとするため (A-B) を  $\text{min}$  レジスタに書き込むとともに、そのフレーム番号  $i$ 、 $j$  を  $\text{min\_i}$ 、 $\text{min\_j}$  に書き込む (s6)。上記の処理を  $i$ 、 $j$  に 1 を加算しながら (s7)、最終フレーム  $\text{end\_i}$ 、 $\text{end\_j}$  について処理するまで繰り返し実行する (s8)。

## 【0037】

全てのフレームについて上記の処理を行ったのち、 $\text{min\_i}$ 、 $\text{min\_j}$  を結合フレームと決定して、結合処理に進む (s10)。このとき、もしフレーム番号  $\text{min\_i}$  がダミーデータ (-1) のままであれば、 $A \geq B$  を満たすフレームが無かったものとして、結合処理に進まずにエラー処理 (s11) に進む。

## 【0038】

図6はMP3エンコードされた分割データの結合処理を説明する図である。この処理では、上記の処理で決定された結合フレームで分割データMP3(1)と分割データMP3(2)を結合する。同図では、MP3(1)の結合フレーム  $\text{min\_i}$  のメインデータビギン (ビット貯蔵値) が160であり、MP3(2)の結合フレーム  $\text{min\_j}$  のメインデータビギン (ビット貯蔵値) が150の場合を示している。

## 【0039】

まず、分割データMP3(2)において、結合フレーム  $\text{min\_j}$  より前に繰り上げられている150サンプルのメインデータを読み出して蓄積しておく。次に、分割データMP3(1)において、結合フレーム  $\text{min\_i}$  のメインデータビギンが存在するフレームから結合フレームの直前 ( $\text{min\_i} - 1$ ) のフレームまでを合成対象フレームとする。この合成対象フレームにおいて、ヘッダ、サイドインフォメーション、フレームサイズ、および、結合フレーム  $\text{min\_i}$  のメインデータビギン以前のメインデータについては、分割データMP3(1)のものをを用いる。そして、そのメインデータにつづけて上記 ( $A - B = \text{min}$ ) のサイズのダミーデータを挿入したのち、上記蓄積しておいた結合フレーム  $\text{min\_j}$  以後の分割データMP3(2)のメインデータをMP3(1)のメインデータエリアに書き込む。

そして、上記合成対象フレームより後の (結合フレーム  $\text{min\_j}$  を含む) フ

レームは分割データMP 3 (2) のものを結合する。

#### 【 0 0 4 0 】

図 7 (A) は、複数の分割データおよび複数のプロセッサ (DSP 15-1 ~ n) を管理する CPU 10 の動作を示すフローチャートである。また、同図 (B) は、分割データ管理テーブル、プロセッサ管理テーブルの更新内容を説明する図である。

同図 (A) において、曲データ (PCMオーディオデータ) が読み込まれると、この曲データを図 3 に示したような方式で特定サイズ毎に切り出して分割データを作成し、HDD 11 に仮記憶する (s 2 1)。このとき各分割データにはそれぞれ属性情報が付与され、この曲データに対応する分割データ管理テーブルが作成される。この分割データ管理テーブルでは、各分割データに対して同図 (B) の 1 0 1 のようなレコードが設定される。s 2 1 で曲データから分割データが切り出されたときは、全ての分割データのステータスは「未処理」である。

#### 【 0 0 4 1 】

データの分割後、ステータスが「未処理」の分割データが無くなったことを s 2 2 で判断するまで、s 2 3、s 2 4 の処理を実行する。s 2 3 ではステータスが「処理なし」の DSP が DSP 15-1 ~ n 中にあるか否かを検索する。「処理なし」の DSP がない場合には、「処理なし」の DSP が生じるまで s 2 2、s 2 3 で待機する。「処理なし」の DSP がある場合には s 2 3 から s 2 4 に進み、この「処理なし」の DSP に対して「未処理」の分割データを転送して MP 3 のエンコード処理を行わせる (s 2 4)。

同時に、この DSP のステータスを「符号化処理中」に書き換えてプロセッサ管理テーブルを更新し (1 1 2)、この分割データのステータスを「符号化中」に書き換えて分割データ管理テーブルを更新する (1 0 2)。

#### 【 0 0 4 2 】

DSP は、CPU 10 から転送された分割データの MP 3 エンコード処理を終了すると、CPU 10 に返送する。CPU 10 は、この MP 3 エンコードされた分割データを HDD 11 に書き込んで仮に記憶するとともに、分割データ管理テーブルのこの分割データのステータスを 1 0 3 に示すように「符号化終了」に書



き換え、プロセッサ管理テーブルのこのDSPのステータスを「空き」に書き換えて111の状態にもどす。

## 【0043】

全ての分割データについて上記処理が実行され、「未処理」の分割データが無くなるとともに、全ての分割データの処理が終了して「符号化処理終了」になると、s25の判断で、上記図4～図6に示した合成処理に進む。

## 【0044】

図7の処理手順では、「処理なし」のDSPに対して分割データのエンコードを実行させているときに、クライアント装置からリクエストがあった場合でも、このDSPまたは他のDSPが「処理なし」になるまで、リクエストに対応するデコード処理（他処理）を受け付けることができないが、デコード処理（他処理）をエンコード処理に優先させ、リクエストがあったときに「処理なし」のDSPがない場合には、エンコード処理を実行しているDSPの処理を中止して、このリクエストに対応するデコード処理を実行させるようにしてもよい。

## 【0045】

このような場合の処理を図8に示す。この図では、エンコード途中にリクエストが発生した場合には、分割データのエンコード処理を途中で中止し、その途中の位置で分割データを処理済部分と未処理部分にさらに分割してCPU10に返送するようにしている。CPU10は、処理済部分を「符号化終了」の分割データとして扱い、未処理部分を「未処理」の分割データとして扱えばよい。

## 【0046】

図8（A）は、クライアント装置からリクエストが発生した場合、すなわち他処理動作要求が発生した場合の動作を示すフローチャートである。他処理動作要求が発生すると、ステータスが「空き」のDSPがあるかをプロセッサ管理テーブルから検索する（s31）。「空き」のDSPがある場合には、他のDSPの処理を中止させる必要がないため、この「処理なし」のDSPに対してこの他処理動作を割り当てるとともに（s34）、このDSPのステータスを「他処理中」にする。

## 【0047】

一方、他処理動作要求が発生したときに「処理なし」のDSPが存在しなかった場合には、現在分割データをMP3にエンコードしているDSP、すなわちステータスが「符号化中」のDSPがあるかを検索する。「符号化中」のDSPもない場合には、いずれかのDSPが他処理（デコード処理）を終了して「処理なし」になるまでs 3 1で待機する。

## 【 0 0 4 8 】

「符号化中」のDSPがあった場合には、そのDSP（複数のDSPがエンコード（符号化）を行っている場合には、そのうちの1つ）のエンコード処理を中止し、DSPからデータを回収する。そして、同図（C）に示すように、この中止までにエンコードしたフレームを境界として、データを再分割する。この再分割は、未処理データを新たな分割データとすることで行う。

## 【 0 0 4 9 】

そして、同図（B）に示すように、分割データ管理テーブルの内容を更新する。エンコード終了分のデータについては「符号化終了」の分割データとしてHDD 1 1に仮記憶するとともに、分割データ管理テーブルの元の分割データのレコードをこれに合わせて書き換える。すなわち、サイズをエンコード済のフレーム数 $m$ に書き換え、ステータスを「符号化終了」に書き換える。そして、新たな分割データをHDD 1 1上に生成するとともに、先頭フレーム番号が「元のデータのスタートフレーム番号（ $frame\ NO$ ）+ $m-ovl$ 」、サイズが「元のデータのサイズ（ $size$ ）- $m+ovl$ 」、ステータスが「未処理」のレコードをこの分割データのレコードとして分割データ管理テーブルに生成する。

## 【 0 0 5 0 】

なお、エンコード済のフレーム数がオーバーラップフレーム数 $ovl$ よりも少ない場合には、分割しても非効率であるため、上記分割処理は行わず、エンコードデータを破棄して、この処理は無かったことにする。

そして、上記処理を中止したDSPについては、そのステータスを「処理なし」に書き換える。こののち、s 3 4に進む。

## 【 0 0 5 1 】

これにより、クライアント装置からのリクエストを内部的な処理であるMP3

エンコードに優先させて実行することができ、オーディオサーバ装置としての機能を低下させることがない。

【 0 0 5 2 】

なお、この実施形態では、PCMオーディオデータをMP3データにエンコードする場合の処理について説明したが、これ以外の形式であっても、エンコード時にエンコード箇所よりも前または後のデータが必要なエンコード方式であれば全て適用可能である。

【 0 0 5 3 】

また、この実施形態では、オーディオサーバ装置を例にあげて説明したが、この発明はどのような装置にも適用することができる。すなわち、他処理はMP3データのデコードに限定されない。

【 0 0 5 4 】

【発明の効果】

以上のようにこの発明によれば、PCMオーディオデータを複数の分割データに分割し、各分割データを複数のプロセッサの空き時間を利用してエンコードするようにしたことにより、プロセッサの利用効率を向上することができるとともに、分割データ単位のエンコード処理であるため、プロセッサの占有時間が短く、その他の処理の妨げになることが少ない。

【 0 0 5 5 】

【図面の簡単な説明】

【図1】 この発明の実施形態であるオーディオサーバ装置のブロック図

【図2】 MP3データのフォーマットを説明する図

【図3】 オーディオサーバ装置のCPUの分割処理を説明する図

【図4】 同CPUの結合フレーム解析処理を説明する図

【図5】 同CPUの結合フレーム解析処理動作を示すフローチャート

【図6】 同CPUの結合処理を説明する図

【図7】 同CPUのDSPおよび分割データ管理動作を示すフローチャート

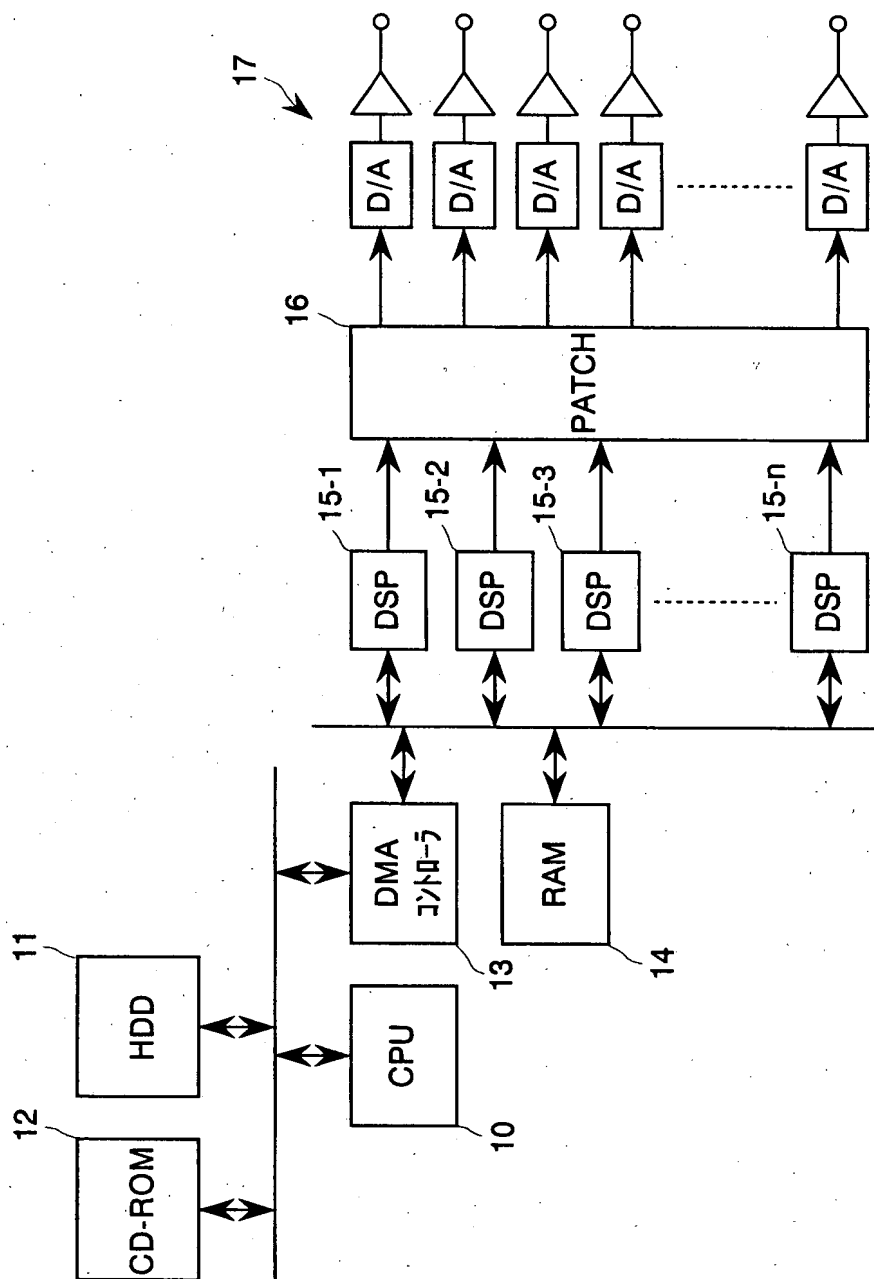
【図8】 同CPUのDSPおよび分割データ管理動作を示すフローチャート

【符号の説明】

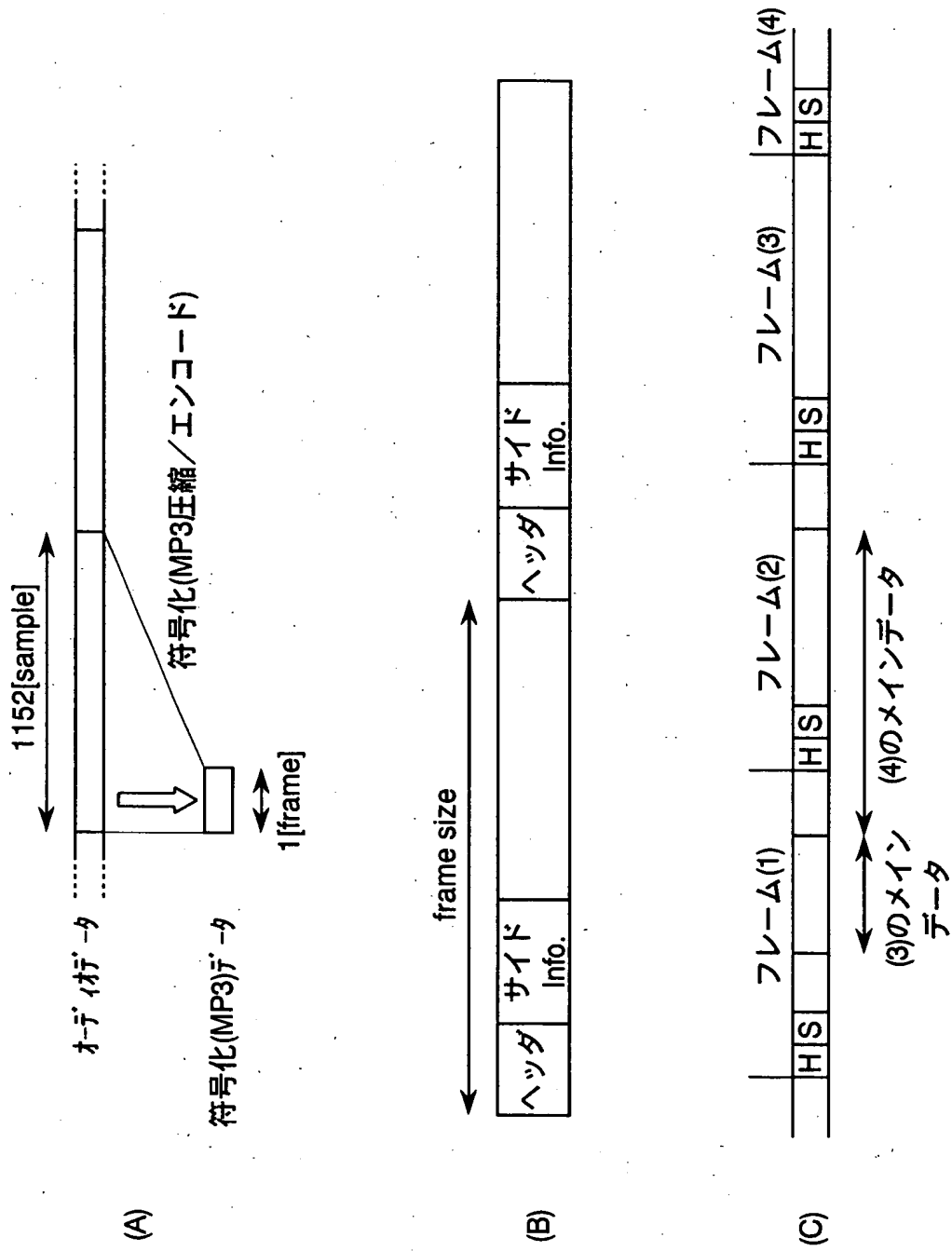
10…CPU、11…HDD、12…CD-ROMドライブ、13…DMAコントローラ、14…RAM、15 (15-1~n) …DSP (プロセッサ)、16…パッチベイ、17…アナログ回路

【書類名】 図面

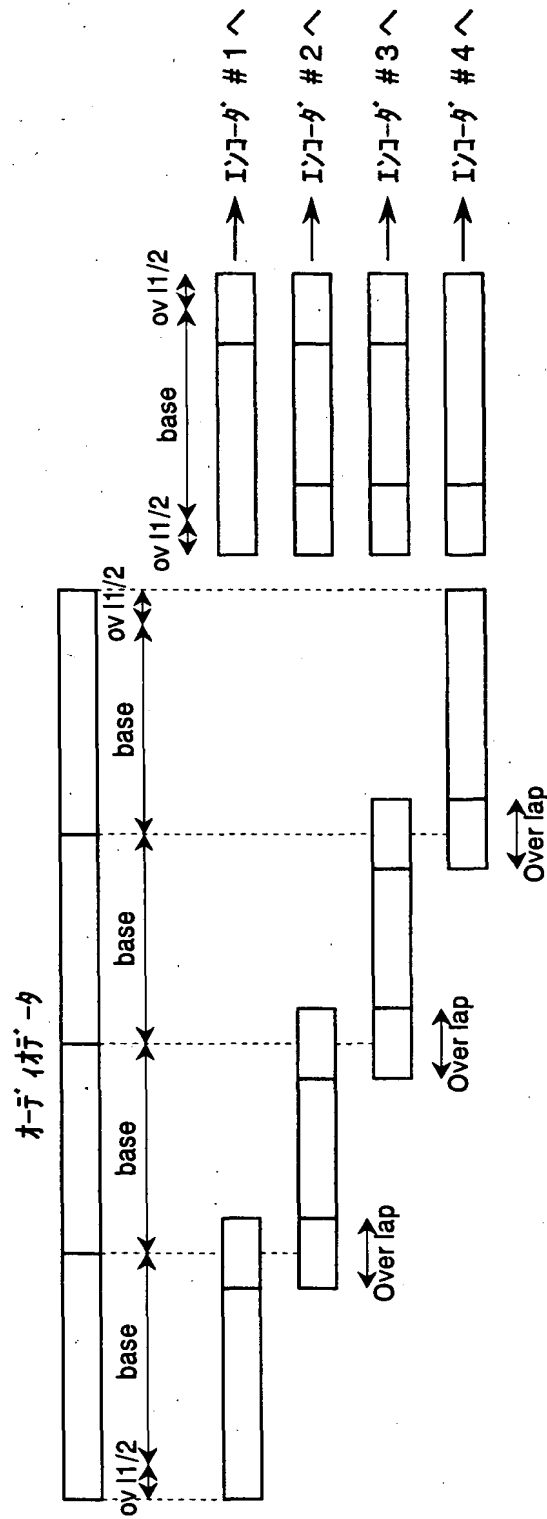
【図 1】



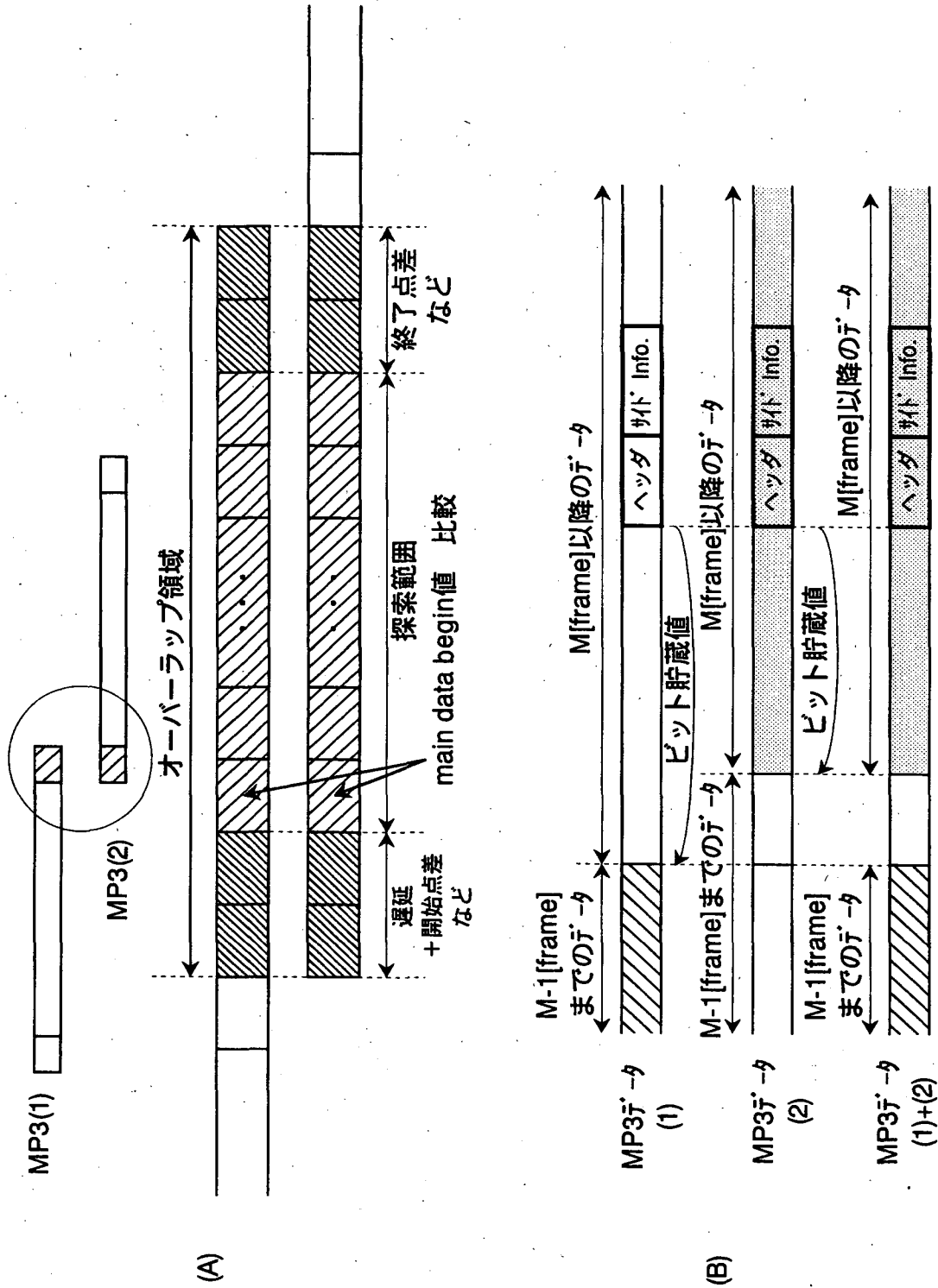
【図 2】



【図 3】

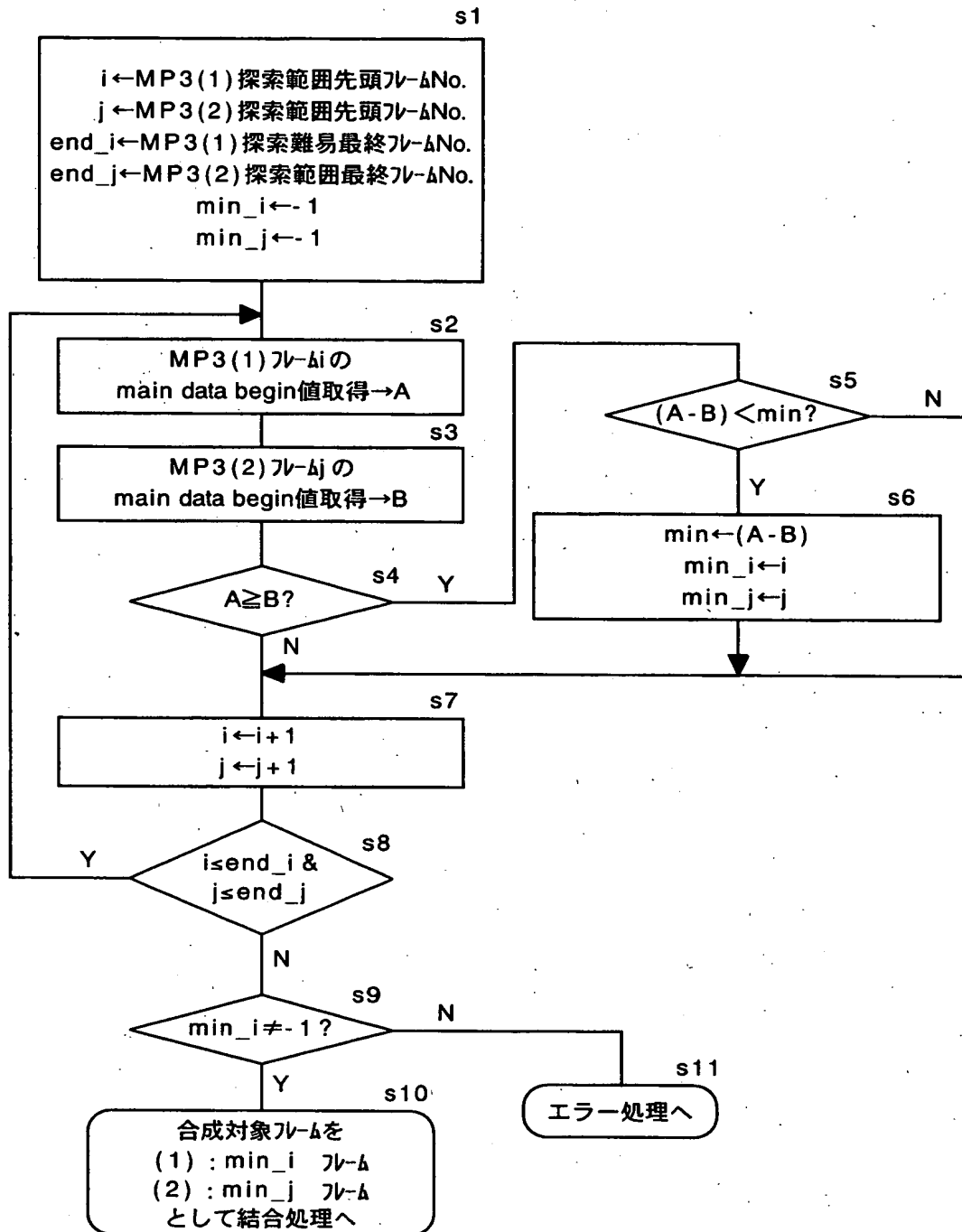


【図 4】

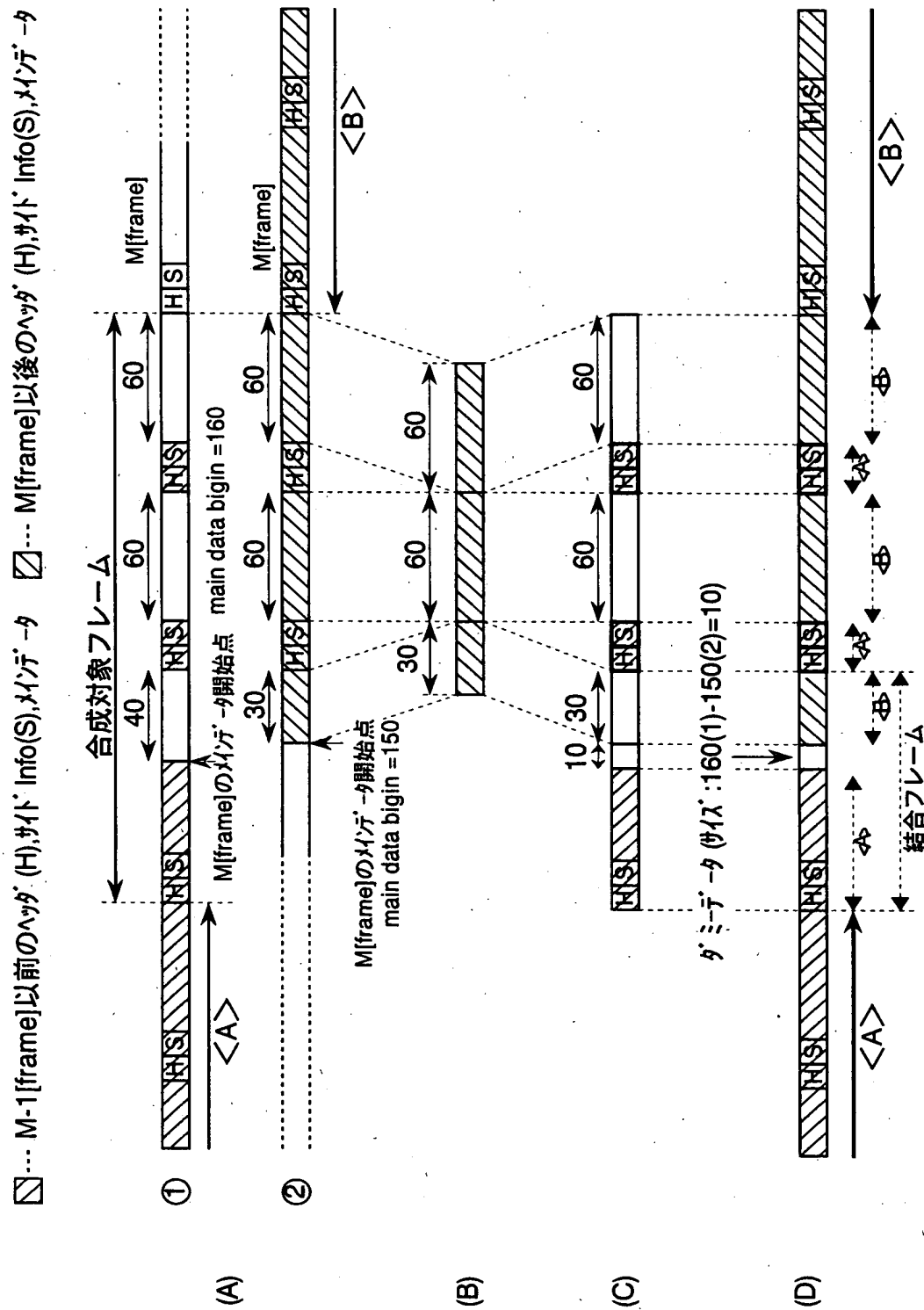




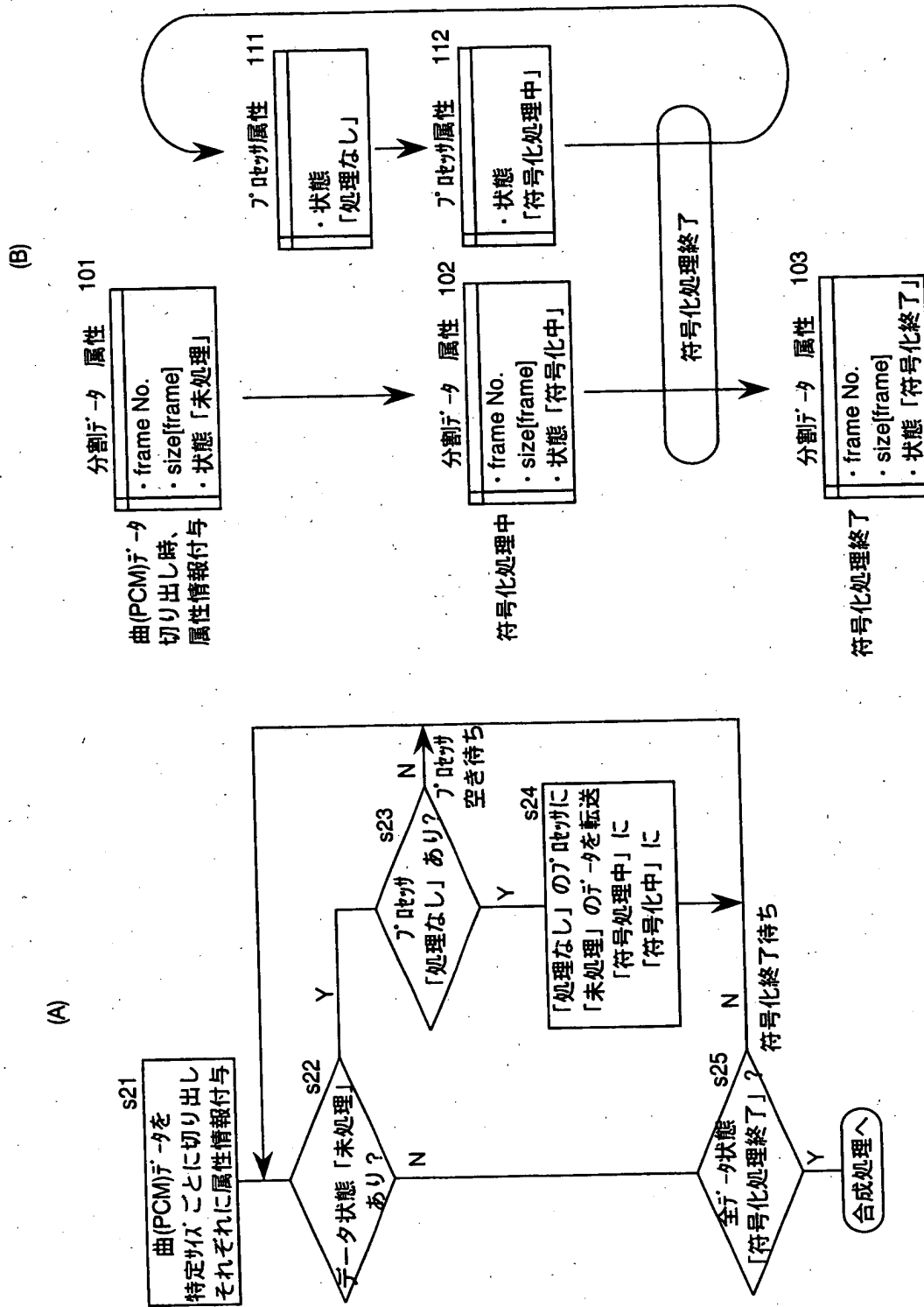
【図 5】



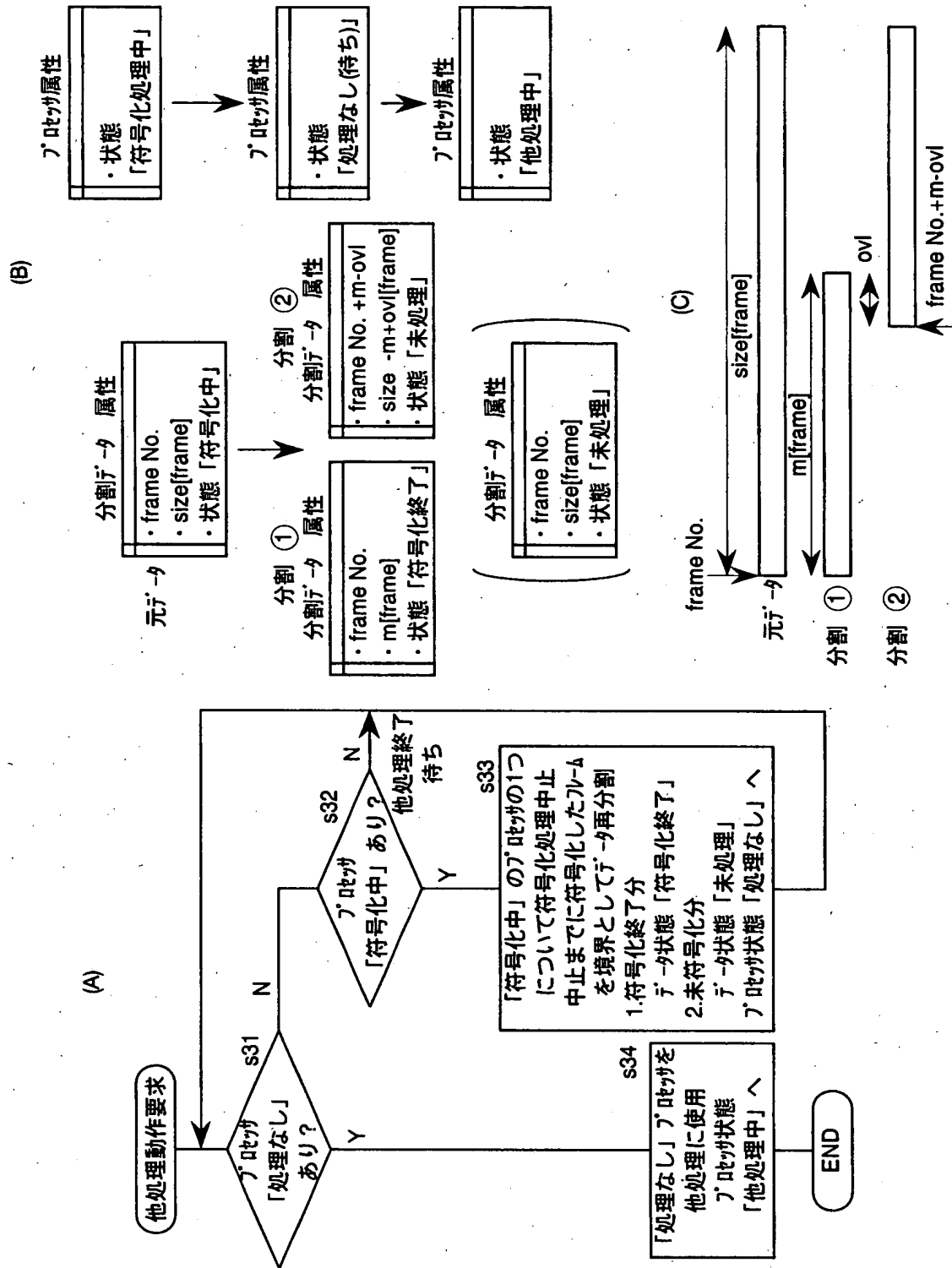
【图 6】



【図 7】



【図 8】



【書類名】 要約書

【要約】

【課題】 オーディオデータのエンコード処理と他の処理とを動的に割り当てながら、効率的にオーディオデータのエンコード処理を行うことができるオーディオデータ処理装置を提供する。

【解決手段】 PCMオーディオデータを複数の分割データに分割し、分割データ単位でエンコード処理を行う。複数のDSPはオーディオ信号をクライアント装置に供給するために、MP3データのデコード処理を実行するものであるが、このデコード処理の空き時間を利用して前記分割データのエンコード処理を行う。分割データ単位のエンコード処理はDSPを占有する時間が短いため、メインの処理であるデコード処理の妨げになりにくい。

【選択図】 図1

出 願 人 履 歴 情 報

識別番号 [000004075]

1. 変更年月日 1990年 8月22日  
[変更理由] 新規登録  
住 所 静岡県浜松市中沢町10番1号  
氏 名 ヤマハ株式会社